

1/8

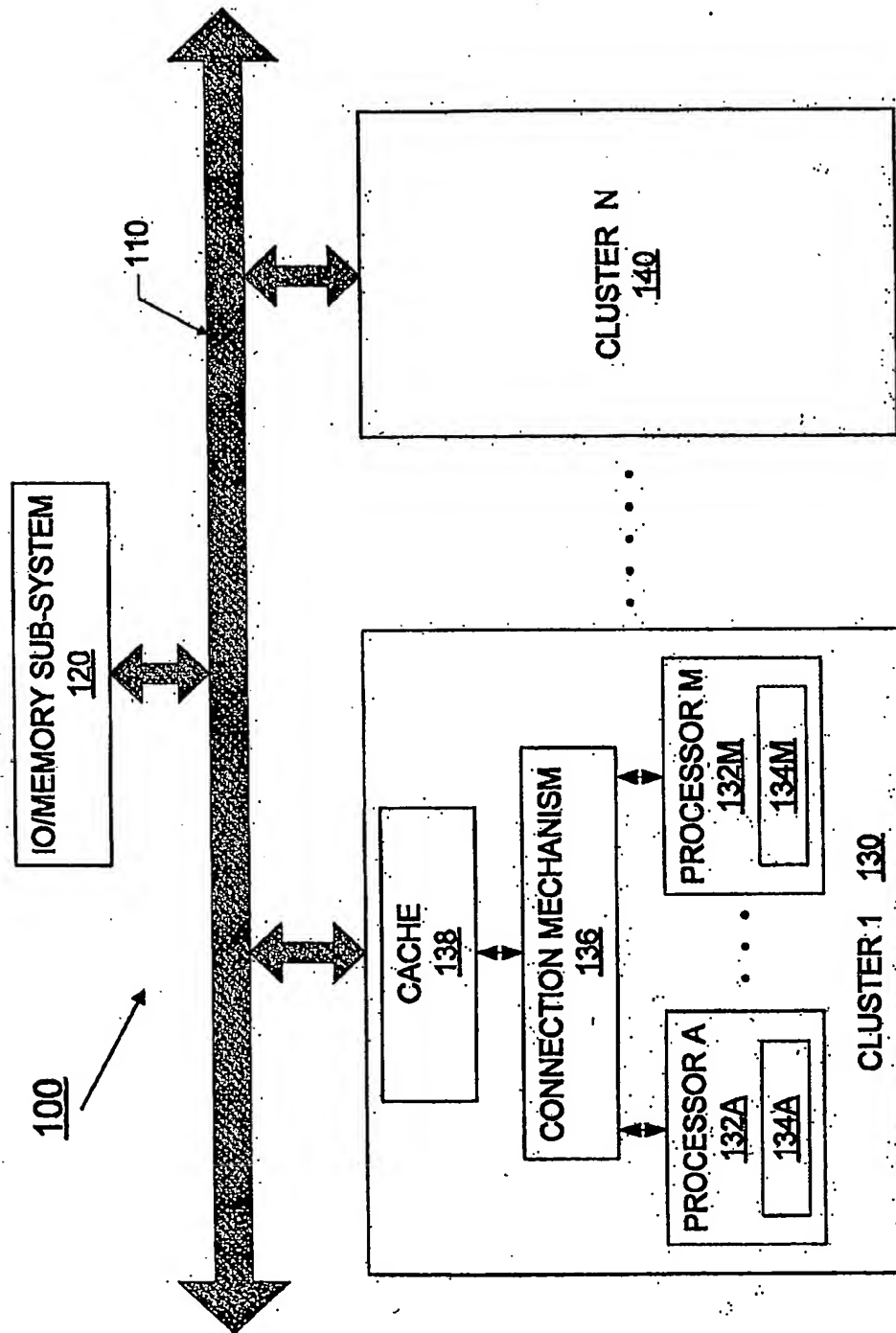


FIGURE 1

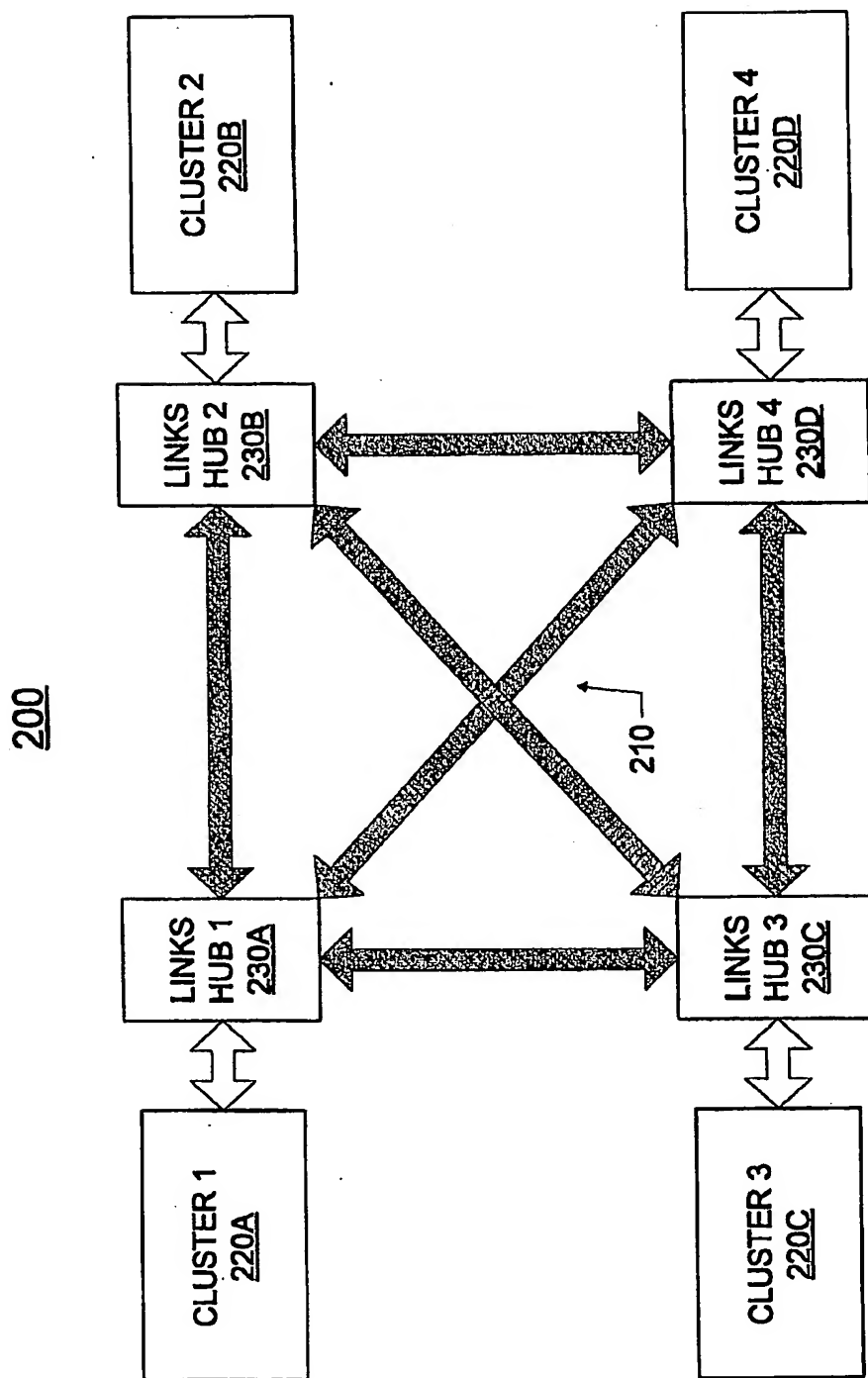


FIGURE 2

3/8

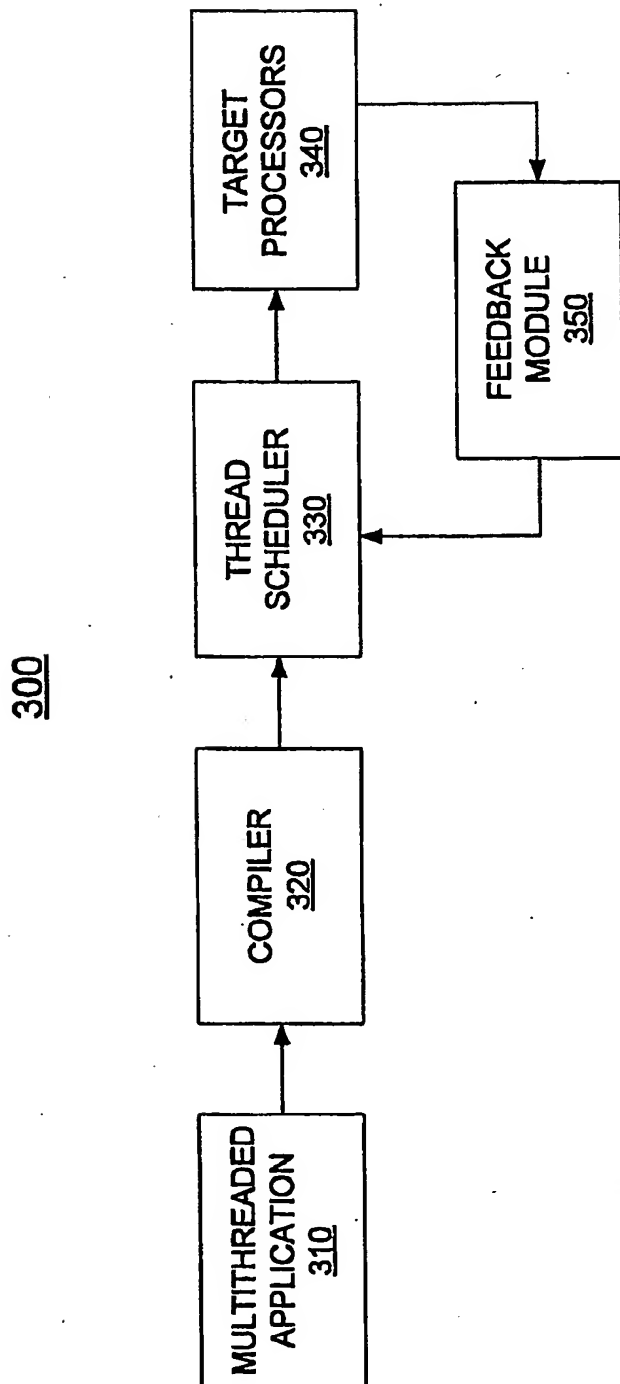


FIGURE 3

4/8

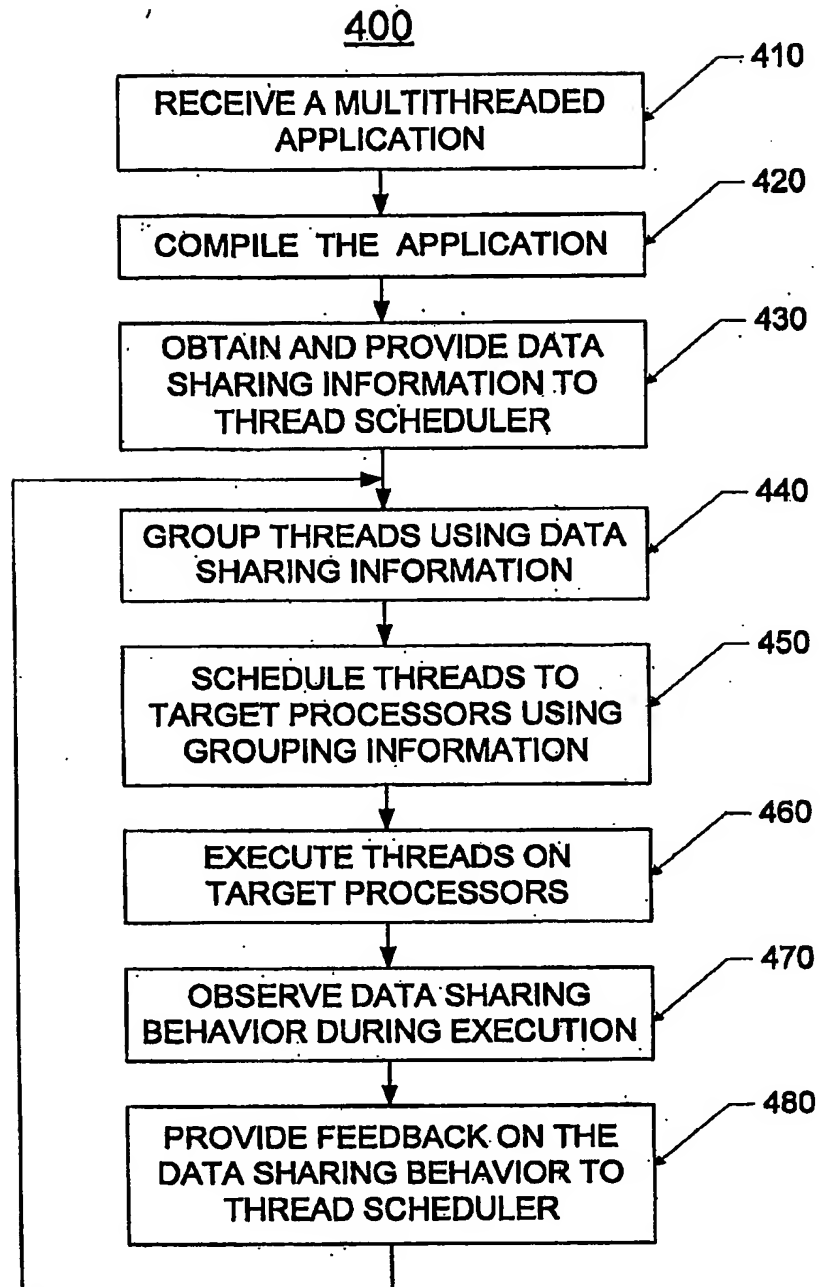


FIGURE 4

600

```
605 void test(List p).  
610 {  
615   #pragma intel omp parallel taskq shared(p) num_threads(4)  
620   {  
625     while (p != NULL)  
630     {  
635       #pragma intel omp task capture private(p)  
640       {  
645         do_work(p);  
650       }  
655       p = p->next;  
660     }  
665   }  
670 }
```

FIGURE 6

500

```
510 void test(List p)  
520 {  
530   while (p != NULL)  
540   {  
550     do_work(p);  
560     p = p->next;  
570   }  
580 }
```

FIGURE 5

700

```

705 __kmpc_serialized_parallel;
710 __kmpc_fork_call ; create a team of threads
715 __kmpc_taskq ; select one thread to be master thread
720 __kmpc_end_taskq ; turn others to worker threads
725 __kmpc_end_serialized_parallel

750 __kmpc_task_buffer ; allocate memory for thunk_t
755 copy captureprivate variable to thunk_t
760 __kmpc_task ; enqueueing of a task
765 __kmpc_taskq_task ; enqueueing the task_thunk
770 __kmpc_end_taskq ; turn into worker thread

```

FIGURE 7

800

```

805 __kmpc_serialized_parallel;
810 __kmpc_fork_call ; create a team of threads
815 __kmpc_team_scheduling ; schedule team of threads on nearby processors
820 __kmpc_taskq ; select one thread to be master thread
825 __kmpc_end_taskq ; turn others to worker threads
830 __kmpc_end_serialized_parallel

850 __kmpc_task_buffer ; allocate memory for thunk_t
855 copy captureprivate variable to thunk_t
860 __kmpc_task ; enqueueing of a task
865 __kmpc_taskq_task ; enqueueing the task_thunk
870 __kmpc_end_taskq ; turn into worker thread

```

FIGURE 8

7/8

900

```
905 int NumThreads; // specify the number of threads
910 int NumThreadsPerGroup; //specify the number of threads per group
915 int GroupNum = NumThreads / NumThreadsPerGroup;
920 #pragma omp parallel for num_threads(GroupNum)
925 {
930     for (int gid = 0; gid < GroupNum; gid++)
935     {
940         #pragma Intel omp parallel taskq num_threads(NumThreadsPerGroup) ordered
           shared(frame)
           {
           {
           #pragma Intel omp ordered
           frame = decoding_video0;
           #pragma Intel omp task captureprivate(frame)
           extract_feature(frame);
           }
           }
           }
945
950
955
960
965
970
975
980 }
```

FIGURE 9

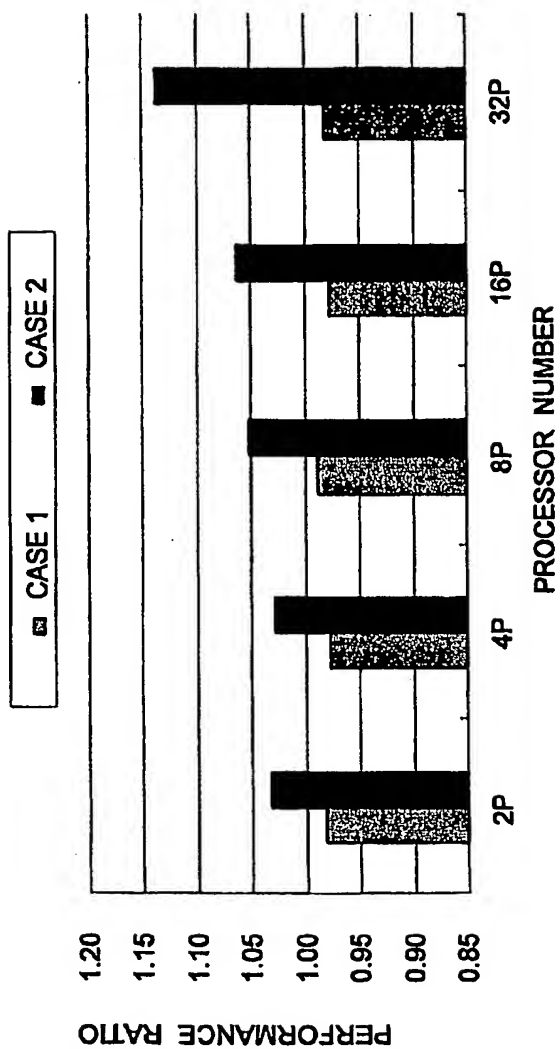


FIGURE 10